PENGGUNAAN HISTORI USER PADA MESIN PENCARI DOKUMEN BERBAHASA INDONESIA

Indriati[1], Heliza Rahmania Hatta[2]
Universitas Brawijaya Malang[1]
Universitas Mulawarman Samarinda[2]
indriati.tif@ub.ac.id[1], heliza_rahmania@yahoo.com[2]

Abstrak. *Banyak mesin pencari informasi yang telah dikembangkan untuk memenuhi semua pencarian user dalam berbagai kebutuhan di dunia nyata. Tetapi mesin pencari informasi tersebut belum terlalu memperhatikan histori user dan akan menampilkan hasil pencarian tidak sesuai dengan kebutuhan user. Dalam penelitian ini, diusulkan sebuah mesin pencari yang mempertimbangkan dokumen histori user pada hasil pencarian yang ditampilkan. Mesin pencari ini menggunakan metode dimana bobot dokumen yang ditampilkan dari hasil pencarian dihitung berdasarkan nilai mean dari bobot dokumen histori user dan bobot dokumen pada pencarian tanpa histori user. Hasil percobaan menghasilkan nilai rata – rata Precision 0,65, Recall 0,83, dan F-Measure 0,68. Metode ini terbukti dapat meningkatkan kinerja pencarian yang disesuaikan untuk setiap user.*

Kata kunci : mesin pencari informasi, histori user, pembobotan

Kemajuan dalam teknologi informasi telah membuat lingkungan bijaksana dan transparan untuk berbagi informasi dalam bentuk konten digital seperti dokumen. Untuk menangani beraneka ragam konten digital, sistem pencarian informasi telah menjadi sebuah isu penting [1]. Pencarian informasi adalah bidang yang menyangkut struktur, analisis, organisasi, penyimpanan, pencarian, dan pengambilan informasi. Pencarian informasi ini memproses kumpulan data dari *rekord* dan meminta informasi, mengidentifikasi dan mendapatkan kembali *rekord* tertentu sebagai respon terhadap permintaan informasi. Proses pengambilan informasi dapat dibuat cerdas untuk membantu user dalam mencari informasi. Beberapa tahun ini banyak yang mengembangkan metode dan mesin pencari informasi yang berhubungan dengan pengambilan informasi. Tujuannya adalah memecahkan masalah informasi yang berlebihan, sehingga dapat memfasilitasi pertukaran dan penggunaan informasi [1]. Maka untuk mengatasi banjir informasi ini, diperlukan memberikan informasi personal kepada *user*.

Dengan variasi informasi yang sangat besar, permasalahan mulai muncul ketika *user* ingin mencari informasi yang sesuai dengan kebutuhannya [2]. Selain itu sistem pencarian informasi yang baik dan metode yang sesuai memungkinkan *user* menentukan secara cepat dan akurat apakah isi dari dokumen yang diterima memenuhi kebutuhannya.

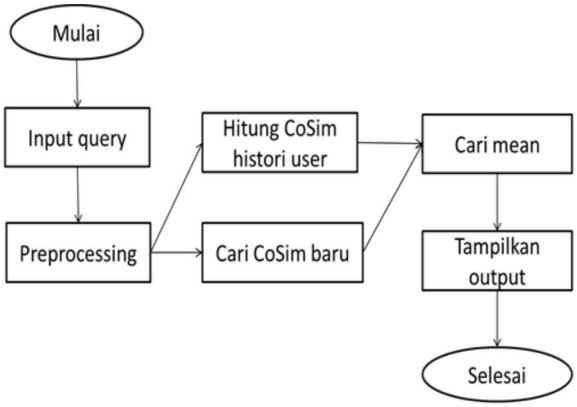
Menurut Giridhar Kumaran dan James Allan [3] bahwa metode *adapting query* berguna untuk memunculkan informasi yang diperkirakan dibutuhkan oleh *user*. Dan metode *query log* berfungsi sebagai histori dari *user* saat melakukan pencarian yang hanya memperhatikan *query* saja[4].

Namun metode – metode tersebut belum terlalu memperhatikan dokumen histori**.** Oleh karena itu, diusulkan sebuah sistem pencarian informasi sesuai dengan histori *user* untuk menampilkan hasil informasi yang tepat. Dimana akan mempertimbangkan dokumen histori *user* pada hasil pencarian dengan menggunakan nilai mean dari bobot dokumen histori user dan bobot dokumen pada pencarian tanpa histori user. Dengan begitu, diharapkan hasil pencarian informasi akan sesuai dengan kebutuhan *user* berdasarkan histori *user* tersebut.

METODOLOGI

Mesin Pencarian Informasi menggunakan Histori User

Pada Gambar 1 memperlihatkan bahwa setelah melakukan proses *preprocessing* yaitu mencari dokumen dengan menggunakan *vector*



Gambar 1. Diagram Alir Sistem

space model, melakukan *stemming* terhadap semua dokumen, melakukan pembobotan dengan *term frequency* dan *inverse document frequency*, kemudian mencari dokumen yang *relevan* dengan menggunakan *cosine similarity* pada tiap dokumen maka akan di hitung nilai *mean* dari bobot dokumen histori user dan bobot dokumen pada pencarian tanpa histori user sehingga menghasilkan bobot baru pada dokumen yang ditampilkan, menggunakan rumus 4.

Vector Space Model

Untuk mengimplementasikan metode – metode klasifikasi dokumen teks, diperlukan suatu transformasi yang dapat mengubah teks – teks *digital* menjadi suatu model yang lebih efisien dan dapat dimengerti sehingga proses analisa dapat dilakukan [1].

Vector space model (VSM) adalah salah satu pendekatan yang paling banyak digunakan dalam merepresentasikan dokumen teks. Pada VSM, setiap dokumen di dalam *database* dan *query user* direpresentasikan oleh suatu *vector multi-dimensi* [5]. Dimensi sesuai dengan jumlah *term* dalam dokumen yang terlibat. pada model ini:

- *Vocabulary* merupakan kumpulan semua *term* berbeda yang tersisa dari dokumen setelah *preprocessing* dan mengandung *t term index*. *Term* - *term* ini membentuk suatu ruang *vector*
- Setiap *term i* di dalam dokumen atau *query j*, diberikan suatu bobot (*weight*) bernilai real w_{ij}.
- Dokumen atau *query* diekspresikan sebagai *vector* dimensi $d_j = (w_1, w_2, ..., w_{tj})$ dan terdapat *n* dokumen di dalam koleksi, yaitu $j = 1, 2, ..., n$.

Terms	Frekuensi
Joni	1
suka	1
bermain	1
mainannya	1
ada	1
banyak	1

Gambar 2. Penggunaan Vector Space Model

Dalam model ini, setiap dokumen *dj* ditransformasikan menjadi suatu vektor [5]:

$$d_j = (w_{1j}, w_{2j}, ..., w_{ij}). \quad (1)$$

Penggunaan VSM ini dapat dilihat pada Gambar 2.

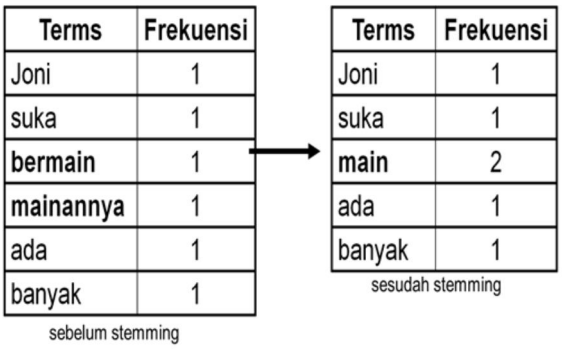
Stemming

Stemming adalah suatu proses pengembalian suatu kata berimbuhan ke bentuk dasarnya (*stem/root*). Sebagai contoh, *stemming* pada kata "mempermainkan" akan menghasilkan kata "main". *Stemming* adalah alat pemrosesan teks dasar yang sering digunakan untuk mendapatkan kinerja yang efektif dan efisien pada *text retrieval* dan *text classification*. Namun, seperti halnya *stopping*, kinerja *stemming* juga bervariasi dan sering tergantung pada domain bahasa yang digunakan [6]. Metode stemming yang digunakan adalah *confix stemmer*.

Dalam morfologi kata Bahasa Indonesia dikenal adanya 3 imbuhan yaitu awalan (*prefix*), sisipan (*infix*), dan akhiran (*suffix*). Dalam metode confix stemmer ini dimulai dengan pembacaan tiap kata pada sampel kemudian mengelompokkan dan mengenkapsulasi imbuhan – imbuhan, termasuk di dalamnya adalah awalan (*prefix*), sisipan (*infix*), akhiran (*suffix*) dan gabungan awalan-akhiran (*confixes*). Untuk penanganan dokumen yang mengandung kata jadian, hanya akan menghilangkan awalan dan akhiran [2]. Sehingga bentuknya menjadi:

[DP + [DP + [DP +]]] Kata Dasar [[+ DS][+ PP][+ P]],

dimana:



Gambar 3. Penggunaan Confix Stemmer

- *Inflection suffixes* : kelompok – kelompok akhiran yang tidak mengubah bentuk kata dasar. Kelompok ini dapat dibagi menjadi dua :
  - *Particle (P)* atau partikel.
  - *Possessive Pronoun (PP)* atau kata ganti kepunyaan.
- *Derivation Suffixes* (*DS*) : kumpulan akhiran asli Bahasa Indonesia yang secara langsung ditambahkan pada kata dasar.
- *Derivation Prefixes (DP)* : kumpulan awalan yang dapat langsung diberikan pada kata dasar murni, atau pada kata dasar yang sudah mendapatkan penambahan sampai dengan 2 awalan.

Penjelasan lebih lanjut dapat dilihat pada Gambar 3.

TFIDF *Weighting*

Bobot setiap *term* dapat direpresentasikan secara binari (*true* atau *false*), frekuensi, atau dengan *term frequency* dan *inverse document frequency* (TFIDF). Metode pembobotan TFIDF adalah metode yang paling umum digunakan untuk mendeskripsikan dokumen dalam *Vector Space Model*, terutama dalam bidang *information retrieval*. Bobot tersebut merupakan perhitungan statistik untuk mengevaluasi bagaimana pentingnya sebuah term terhadap sebuah dokumen atau koleksi. Tingkat kepentingan bertambah secara proporsional terhadap jumlah kemunculan sebuah *term* dalam sebuah dokumen namun diimbangi dengan frekuensi kemunculannya dalam koleksi dokumen.

Metode TFIDF telah menunjukkan performa yang lebih baik jika dibandingkan dengan metode binari dan frekuensi [3], yang dinyatakan sebagai berikut:

$$w_{ij} = tf_{ij} . \log_2 \left(\frac{N}{df_i} \right), \quad (2)$$

dimana w_{ij} adalah bobot *term i* pada dokumen *j*, tf_{ij} adalah frekuensi *term i* pada dokumen *j*, *N* adalah jumlah dokumen yang diproses dan df_j adalah jumlah dokumen yang memiliki *term i* di dalamnya.

Cosine Similarity

VSM dan pembobotan TFIDF digunakan untuk merepresentasikan nilai numerik dokumen sehingga kemudian dapat dihitung kedekatan antar dokumen. Semakin dekat dua vektor di dalam suatu VSM maka semakin mirip dua dokumen yang diwakili oleh vektor tersebut. Kemiripan antar dokumen dihitung menggunakan suatu fungsi ukuran kemiripan (*similarity measure*).

Salah satu ukuran kemiripan teks yang popular [7] adalah *cosine similarity* (*CoSim*) selain itu dikarenakan lebih sesuai untuk digunakan di VSM. Ukuran ini menghitung nilai *cosinus* sudut antara dua vektor. Jika terdapat dua vektor dokumen *dj* dan *query q*, serta *t term* diekstrak dari koleksi dokumen maka nilai *cosinus* antara *dj* dan *q* didefinisikan sebagai berikut:

$$similarity(\vec{d}_j, \vec{q}) = \frac{\vec{d}_j . \vec{q}}{|\vec{d}_j| . |\vec{q}_j|} = \frac{\sum_{i=1}^{t} (w_{ij} . w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^2 . \sum_{i=1}^{t} w_{iq}^2}}. \quad (3)$$

CoSim ini berlaku untuk dokumen histori user maupun pada dokumen hasil pencarian. Sehingga untuk melihat pengaruh antara dokumen histori *user* dan dokumen hasil pencarian tanpa histori user digunakan *mean* untuk kedua dokumen tersebut dan dapat dituliskan sebagai berikut:

$$Mean = \frac{similarity\,(histori\,user) + similarity\,(umum)}{2}, \quad (4)$$

dimana *similarity* (histori *user*) adalah *CoSim* dokumen histori *user* dan *similarity* (umum) adalah *CoSim* dokumen hasil pencarian tanpa user histori.

Histori *User*

Histori *user* dalam sistem ini memiliki struktur yang tersimpan dalam basis data mengenai *user*, dokumen yang pernah dibuka oleh *user* serta bobot dokumen tersebut. Selain itu, histori *user* memiliki struktur multi – lapisan hirarki. Lapisan atas adalah lapisan yang berisi hasil pencarian dokumen yang dipilih oleh *user*. Lapisan bawah adalah hasil

pencarian lapisan yang berisi hasil yang dipilih oleh user, dapat dilihat pada Gambar 4.

Tabel 1. Jumlah Dokumen

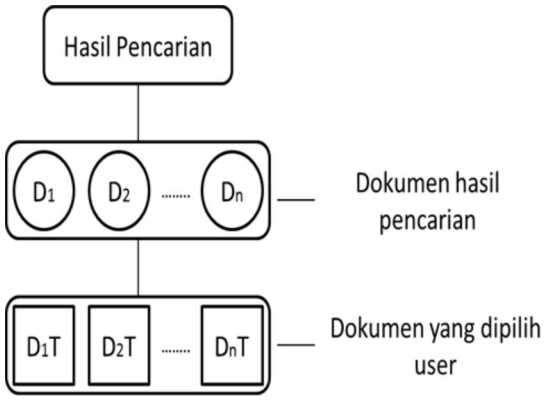
Kategori	Jumlah Dokumen
Nasional	105
Regional	106
Internasional	104
Metropolitan	106
Bisnis dan Ekonomi	101
Olahraga	110
Sains dan Teknologi	90
Edukasi	109
Pariwisata	101
Total	932

Dengan mempertahankan struktur hirarkis, sistem dapat lebih efektif menangani kepentingan *user* yang terus menerus berubah [8]. Untuk mengatasi kepentingan *user* yang sering berubah, sistem menyediakan hasil pencarian yang lebih pribadi. Sistem ini menemukan dokumen yang pernah dicari oleh *user*, kemudian mengembalikan hasil pencarian baru yang merupakan gabungan antara histori *user* dengan dokumen baru yang relevan apabila dicari tanpa histori *user*, setelah dilakukan penghitungan *mean* terhadap dua dokumen tersebut. Perhatikan Gambar 5.

Tabel 2. *Retrieve* dan *Relevant*

	Relevant	*Not Relevant*
Retrieved	TP	FP
Not Retrieved	FN	TN

Ket: TP = *True Positif*, FP = *False Positif*, FN = *False Negatif*, TN = *True Negatif*



Gambar 4. Histori User

Apabila dokumen yang tampil pernah dicari oleh *user* maka hasil *mean* pembobotan akan lebih besar pada pembobotan dokumen histori *user* sedangkan dokumen yang tampil tidak pernah *user* cari maka hasil *mean* pembobotan akan lebih besar pada dokumen baru.

HASIL DAN PEMBAHASAN

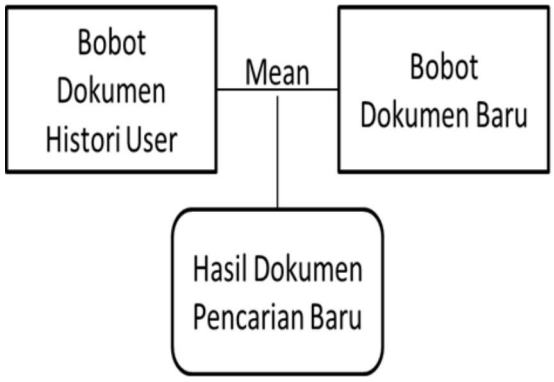
Dokumen yang digunakan adalah *corpus* berita *online* berbahasa Indonesia yang didapatkan dari *www.kompas.com*. Berita yang diunduh berdasar kategori yang telah ditetapkan oleh *www.kompas.com*. Antara satu kategori dengan kategori lainnya memiliki jumlah dokumen training yang berbeda. Spesifikasi jumlah dokumen untuk setiap kategori dapat dilihat pada Tabel 1, bahwa jumlah keseluruhan dokumen adalah 932 yang terdiri dari 9 kategori. Dokumen – dokumen tersebut telah dilabeli kategori sesuai dengan kategori yang diberikan oleh situs berita *www.kompas.com*.

Pelaksanaan uji coba ini menggunakan rumus *precision*, *recall* dan *F-Measure* dengan pendekatan dokumen yang di*retrieve* dan *relevan* seperti pada Tabel 2. Tabel tersebut menunjukkan beberapa *item* yang akan digunakan untuk menghitung *Precision*, *Recall*, dan *F-Measure* dengan rumus sebagai berikut:

Precision (P) = TP / (TP + FP), (5)

Recall (R) = TP / (TP + FN) (6)

F-Measure (F) = 2 * P * R / (P + R) (7)



Gambar 5. Hasil Pencarian

Dari hasil uji coba, didapatkan nilai *Precision*, *Recall*, dan *F-Measure* yang dapat dilihat pada Tabel 3. Dapat dlihat bahwa nilai *F-Measure* pada kategori Metropolitan dan Nasional mencapai nilai 1,00 dikarenakan dokumen yang di tampilkan dari hasil pencarian sedikit. Hal ini terjadi karena *query* yang dimasukkan spesifik terhadap kategori

dokumen. Sedangkan pada kategori Olahraga memiliki nilai *F-Measure* kecil karena *query* yang dimasukkan terdiri dari dua kata yang kurang spesifik terhadap kategori.

SIMPULAN

Berdasarkan uji coba yang telah dilakukan, hasil pencarian menghasilkan dokumen yang pernah dibuka oleh *user* maka dokumen akan berada diperingkat atas karena mempunyai bobot lebih besar dibandingkan dengan dokumen yang tidak pernah dibuka oleh user. Hal ini menunjukkan pengaruh histori user terhadap hasil pencarian dari mesin pencari dengan berubahnya urutan dokumen yang ditampilkan. Selain itu didapatkan juga nilai rata – rata Precision 0,65, Recall 0,83, dan F-Measure 0,68.

Pada metode yang diusulkan ini, dapat dibuktikan bahwa histori *user* juga berperan dalam menentukan hasil pencarian yang sesuai dengan kebutuhan *user*, namun belum optimal dalam menampilkan hasil pencarian yang sesuai dengan *profil user*. Sehingga untuk ke depannya, diharapkan dapat mengkolaborasikan antara histori *user* dengan *profil user* agar hasil pencarian informasi sangat sesuai dengan kebutuhan user.

Tabel 3. Hasil Uji Coba

Kategori	*Precision*	*Recall*	*F-Measure*
Bisnis & Ekonomi	0,40	1,00	0,57
Edukasi	0,91	1,00	0,95
Internasional	0,91	1,00	0,95
Metropolitan	1,00	1,00	1,00
Nasional	1,00	1,00	1,00
Olahraga	0,03	0,50	0,06
Pariwisata	1,00	0,80	0,89
Regional	0,33	0,20	0,25
Sains & Teknologi	0,29	1,00	0,45
Rata - rata	0,65	0,83	0,68

Dari hasil penelitian dapat dibuktikan bahwa dokumen histori user dapat dipertimbangkan pada mesin pencarian informasi agar hasil pencarian yang sesuai dengan kebutuhan user berada pada peringkat atas dari daftar dokumen yang dihasilkan mesin pencari. Metode ini terbukti dapat meningkatkan kinerja pencarian yang disesuaikan untuk setiap user.

DAFTAR PUSTAKA

[1] Ming-Yen Chen, Hui-Chuan Chu, Yuh-Min Chen. 2010. "Developing a semantic-enable information retrieval mechanism". Elsevier.

[2] Aini Rachmania Kusumaagama Fuddoly, Agus Zainal Arifin. 2011. "Klasifikasi Kategori dan Identifikasi Topik pada Artikel Berita Berbahasa Indonesia". Institut Teknologi Sepuluh Nopember.

[3] Giridhar Kumaran, James Allan. 2007. "Adapting information retrieval systems to user queries". Elsevier.

[4] Milad Shokouhi, Justin Zobel, Saied Tahaghoghi, Falk Scholer. 2006. "Using query logs to establish vocabularies in distributed information retrieval". Elsevier.

[5] Polettini, Nicola. 2004. "The Vector Space Model in Information Retrieval - Term Weighting Problem".

[6] Mahendra, I Putu Adhi Kerta. 2008. "Enhanced Confix Stripping Stemmer And Ants Algorithm For Classifying News Document In Indonesian Language". The 5th International Conference on Information & Communication Technology and Systems ISSN 2085-1944.

[7] Tata, Sandeep, Patel M, Jignesh. 2007. "Estimating he Selectivity of *TFIDF* based Cosine Similarity Predicates". Sigmod Record December 2007 Vol 36 No. 4.

[8] Hochul Jeon, Taehwan Kim, Joongmin Choi. 2010. "Personalized Information Retrieval by Using Adaptive User Profiling and Collaborative Filtering". Advances in Information Sciences and Service Sciences.